**Assignment 7**

**Tanmay Parmar**

**300872344**

**MainWindow.xaml**

<Window x:Class="Assignment7.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:Assignment7"

mc:Ignorable="d"

Title="MainWindow" Height="340" Width="350" Background="Beige">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Canvas Grid.Column="1" Margin="0" Name="paintCanvas"

Background="White" MouseMove="paintCanvas\_MouseMove"

MouseLeftButtonDown="paintCanvas\_MouseLeftButtonDown"

MouseLeftButtonUp="paintCanvas\_MouseLeftButtonUp"

MouseRightButtonDown="paintCanvas\_MouseRightButtonDown"

MouseRightButtonUp="paintCanvas\_MouseRightButtonUp"/>

<!--Create StackPanel-->

<StackPanel Margin="3">

<!-- GroupBox For color-->

<GroupBox Grid.ColumnSpan="1" Header="Color" Margin="3"

HorizontalAlignment="Stretch" VerticalAlignment="Top">

<StackPanel Margin="3" HorizontalAlignment="Left" VerticalAlignment="Top">

<!--RadioButtons-->

<RadioButton Name="redRadioButton" Margin="3"

Checked="redRadioButton\_Checked">

Red

</RadioButton>

<RadioButton Name="blueRadioButton" Margin="3"

Checked="blueRadioButton\_Checked">

Blue

</RadioButton>

<RadioButton Name="greenRadioButton" Margin="3"

Checked="greenRadioButton\_Checked">

Green

</RadioButton>

<RadioButton Name="blackRadioButton" Margin="3"

Checked="blackRadioButton\_Checked" IsChecked="True">

Black

</RadioButton>

</StackPanel>

</GroupBox>

<GroupBox Header="Size" Margin="3">

<StackPanel Margin="3">

<RadioButton Name="smallRadioButton" Margin="3"

Checked="smallRadioButton\_Checked" IsChecked="True">

Small

</RadioButton>

<RadioButton Name="mediumRadioButton" Margin="3"

Checked="smallRadioButton\_Checked">

Medium

</RadioButton>

<RadioButton Name="largeRadioButton" Margin="3"

Checked="largeRadioButton\_Checked">

Large

</RadioButton>

</StackPanel>

</GroupBox>

<Button Height="23" Width="75" Name="undoButton" Margin="3,10,3,3"

Click="undoButton\_Clicked">Undo</Button>

<Button Height="23" Width="75" Name="clearButton" Margin="3,10,3,3"

Click="clearButton\_Clicked">Clear</Button>

</StackPanel>

</Grid>

</Window>

**MainWindow.xaml.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Assignment7

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

private int diameter = 8;

private Brush brushColor = Brushes.Black;

private bool shouldErase = false;

private bool shouldPaint = false;

private enum Sizes

{

SMALL =4,

MEDIUM = 8,

LARGE = 10

}

public MainWindow()

{

InitializeComponent();

}

//paints a circle on the canvas

private void PaintCircle(Brush circleColor, Point position)

{

Ellipse newEllipse = new Ellipse();

newEllipse.Fill = circleColor;

newEllipse.Width = diameter;

newEllipse.Height = diameter;

//set ellipse's position

Canvas.SetTop(newEllipse, position.Y);

Canvas.SetLeft(newEllipse, position.X);

paintCanvas.Children.Add(newEllipse);

}

private void paintCanvas\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)

{

shouldPaint = true;

}

private void paintCanvas\_MouseLeftButtonUp(object sender, MouseButtonEventArgs e)

{

shouldPaint = false;

}

private void paintCanvas\_MouseRightButtonDown(object sender, MouseButtonEventArgs e)

{

shouldPaint = true;

}

private void paintCanvas\_MouseRightButtonUp(object sender, MouseButtonEventArgs e)

{

shouldPaint = false;

}

private void paintCanvas\_MouseMove(object sender, MouseEventArgs e)

{

if (shouldPaint)

{//draw a circle of selected color, at current location

Point mousePosition = e.GetPosition(paintCanvas);

PaintCircle(brushColor, mousePosition);

}

if (shouldErase)

{

Point mousePosition = e.GetPosition(paintCanvas);

PaintCircle(paintCanvas.Background, mousePosition);

}

}

private void redRadioButton\_Checked(object sender, RoutedEventArgs e)

{

brushColor = Brushes.Red;

}

private void blueRadioButton\_Checked(object sender, RoutedEventArgs e)

{

brushColor = Brushes.Blue;

}

private void greenRadioButton\_Checked(object sender, RoutedEventArgs e)

{

brushColor = Brushes.Green;

}

private void blackRadioButton\_Checked(object sender, RoutedEventArgs e)

{

brushColor = Brushes.Black;

}

private void smallRadioButton\_Checked(object sender, RoutedEventArgs e)

{

diameter = (int)Sizes.SMALL;

}

private void mediumRadioButton\_Checked(object sender, RoutedEventArgs e)

{

diameter = (int)Sizes.MEDIUM;

}

private void largeRadioButton\_Checked(object sender, RoutedEventArgs e)

{

diameter = (int)Sizes.LARGE;

}

private void undoButton\_Clicked(object sender,RoutedEventArgs e)

{

int count = paintCanvas.Children.Count;

if(count>0)

{

paintCanvas.Children.RemoveAt(count - 1);

}

}

private void clearButton\_Clicked(object sender, RoutedEventArgs e)

{

paintCanvas.Children.Clear();

}

}

}

**ScreenShot**

